UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/763,512	01/23/2004	Christopher Lapkowski	CA920030044US1	7220

7590 08/14/2007

Leslie A. Van Leeuwen
International Business Machines Corporation
Intellectual Property Law Department
11400 Burnet Road
Austin, TX 78758

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
08/14/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/763,512	LAPKOWSKI, CHRISTOPHER	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☐ Responsive to communication(s) filed on *07 June 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-23* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-23* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 6/7/07.

As indicated in Applicant's response, claims 1-23 have been amended. Claims 1-23 are pending in the office action.

Claim Objections

2. Claims 1, 9, 16, 20 are objected to because of the following informalities: the phrase recited as 'manner that said register spills can be **loaded back** into said registers' does not appear a proper syntax because it does not establish any reasonable teaching that a prior register loading (to some registers) have been accomplished for a loading back to 'said registers' to make some sense. The language of the claim only yields realization of rewriting based on some determination, and therefrom, configuring storage; but all this does not entail a prior loading for the use of 'loaded back' to be a proper syntax; that is, the underlying semantic of a *load back* without a antecedent basis cannot be accepted. If the Specifications in view of its inconsistency (Note: Specifications at page 2 depicts *symbolic registers ... to be loaded back* to the registers in parallel) does not remedy to this load **back** (of *register spills*) impropriety, it is urged that the Specifications (e.g. pg. 3-5: *register spills* ... can be *loaded back*) be corrected (without introducing new matter) in order to justify how a loading can be construed as load back with emphasis in the term 'back' and also the nature of what has been previously loaded off and subsequently loaded back. The term 'back' will be given no patentable weight in the course of examining the merits.

Appropriate correction is required.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-23 are rejected under 35 U.S.C. 103(a) as being unpatentable over Kolson et al., 'A Method for Register Allocation to Loops in Multiple Register File Architectures', Proceedings of IPPP's 96, 1996, pp. 28-33 (hereinafter Kolson), further in view of Kahle et al, USPN: 5,867,684 (hereinafter Kahle).

As per claim 1, Kolson discloses a method of handling register spills in a CPU having multiple bank registers, comprising:

determining that register spill instructions in spill code generated by a register allocator can be associated with each other (e.g. *register allocation ... analysis variable accesses ... concurrent accesses* –sec 3, pg. 29, R col; *require memory accesses when updated or necessary for computation* – pg. 28, R col. – Note: register allocation unit analyzing what accesses can be made concurrent reads on spill instruction possibly associated with each other with extension and use of multiple registers – see sec 5.1: register classes -pg. 31) ,

based on the determining, rewriting said register spill instructions (e.g. *must be loaded from memory* – Figure 4, pg. 31; sec 5.2, pg. 31: transfer value into ... registers --- NOTE: Extending the algorithm from standard loop register assignment to using register classes for variable accesses via reg/var mappings to selectively *loading from* (fills) memory and *register storing* (spills) reads on a form of rewriting spills with respect to conventional loop register

assignment – e.g. sec 2, pg. 29, *loop iteration ... loads and stores , spills* - L col -- prior to the extension algorithm – see Fig. 4, pg. 31);

based on said rewritten register instruction, configuring storage of associated register spills in memory in such a manner that said register spills can be 'loaded back' into said registers (Note: the update of register for runtime via extension algorithm discloses a *load* back – see *must be loaded from memory* – Figure 4, pg. 31 – Note: *loaded back* treated as variable values loaded from memory/stack into registers).

However, Kolson does not explicitly disclose a CPU having parallel registers; nor does Kolson disclose rewriting said spill instructions as a parallel register spill instruction, and based upon which configuring memory spills to enable loading 'back' said register spills into said parallel register in parallel.

Kolson, however, discloses a architecture having multiple register implementation disposed as banks of General purpose registers and Auxiliary registers to support concurrent demand of load/store operations for a iterative loops such that a consolidated register file architectural approach facilitates concurrent read and writes for live variables (see sec. 2, pg. 29: L col. Multiple Register Files; *performed in parallel* - sec. 3, L col.; *register banks* - sec: 6.1, Fig. 5, pg. 32).

Another alternative of using a multiple register operation as suggested in parallel read/write in Kolson's register extension approach (see pg. 28, R col) is further disclosed by Kahle's architecture. Kahle teaches using multiple register and multi-register architecture (see col. 2, line 63 to col. 3 line 56) for load/store operations (see Fig. 1-3; SUMMARY, col. 1-2 – Note: multi-register instruction reads on CPU having parallel registers instruction) to address

pipeline or complex floating-point/iterative operations and to boost performance, wherein special buffers (e.g. col. 3; Fig. 1) can act temporary storage for a combined set of load/store (which is analogous to intermediately rearranging spills as transitional instructions subsuming two sequential spills into similar colored register class as taught by Kolson – see Kolson: pg. 29, sec 2-3-- prior to the actual spilling).

Based on the structure of multiple banks of registers for parallel accesses in Kolson (see *register allocation ... analysis variable accesses ... concurrent accesses* –sec 3, pg. 29, R col; Table 4, sec 6.4, 6.5, pg. 33) and Kolson's desirability to exploit maximum architecture register availability to minimize spill code, it would have been obvious for one of ordinary skill in the art at the time the invention was made to apply a pipeline multi-register and parallel multi-processing unit architecture by Kahle (*) to support the extended register banks and concurrent access register allocation as endeavored by Kolson's extended algorithm in order to achieve (i) rewriting spill instructions as one intermediate architectural parallel register instruction and (ii) based upon such rewritten intermediate rearrangement, enabling loading/spill instructions into said parallel registers.

In light of the suggested teachings such as (a) Kolson's banks, register classes for extending concurrent register spilling algorithm, and (b) the suggested intermediate rewriting of spill instructions by Kahle's pipeline floating-point multiple load within a parallel execution units architecture, one skill in the art -- at the time where fast and complex architecture operating with extended words instruction sets exist -- would be motivated to do so (refer to *) because parallel processing units architecture as in Kahle's, equipped with specific instruction (multi-register instruction) making use of a disposition of registers as suggested in Kolson's parallel use

of banks to effectuate one instruction operating as parallel loading upon multiple registers as in Kahle's would further alleviate performance bandwidth and resources of runtime; and that would fit with Kolson's extended register endeavor (see Kolson: Abstract pg. 28; see Kahle: col. 1, lines 44-52).

As per claim 2, Kolson (in view Kahne) discloses wherein said parallel register CPU architecture comprises a first register set and a second register set, and determining whether two register spill instructions can be paired (e.g. *General purpose registers and Auxiliary registers* - sec. 5.2, pg. 31; Fig. 5, pg. 32 – NOTE : the loop iteration for enabling floating-point multiflow by multi-banks by Kolson -- see *distribute ...available registers* - R col. pg 28--signifies a floating-point operation requiring parts thereof to be distributed via a pairing by which variable access can be using two register -- this is further illustrated in Kahne: see col. 5-6).

As per claim 3, Kolson discloses two register spill instructions are in a basic block within said spill code (see sec. 4.1, pg. 29-30).

As per claim 4, Kolson (in view of Kahle) discloses by virtue of claim 2 and the obviousness rationale in claim 1 (using multiple register in one load instruction) discloses determining whether said two register spill instructions relate to matching register locations in each of said first register set and said second register set.

As per claim 5, Kolson (in view of Kahle) discloses determining whether any intervening instructions at all exist between said register spill instructions modify any of said register spill instructions (see *new_mapps, curr_maps iteration boundaries* – Fig. 3, pg. 30; Fig. 4, pg. 31 – Note: checking at *iteration boundaries* reads on ensuring that only the proper

mappings are solely within a determined scope, i.e. no out-of-scope data accessing, no intervening dependency).

As per claim 6, Kolson does not explicitly disclose first allocating space on a memory stack to all paired register spills, then allocating space on said memory stack for any remaining register spills. But the use of memory stack in conjunction with variable loading and register allocation in view of spill code optimization was a known concept at the time the invention was made. It is well-known concept to use memory/register stack to provide for variables loading in context switch, complex iterations or nested procedure calls or floating-point arithmetics; hence based on the loop iteration and register consolidation by Kolson (e.g. *floating-points ... distribute ...available registers* - R col. pg 28—Note: floating-point operation with registers distributed usage suggesting stack for holding floating point operands), it would have been obvious for one skill in the art to accommodate the loading instruction in a parallel register concurrent execution so that a register stack be used and spills code be loaded back according to the sequence needed; i.e. allocate first in stack register spills code in order as called for by the stack because this would enable the data needed for the procedure to be available in the correct sequence.

As per claim 7, Kolson does not disclose allocating space on said memory stack such that paired register spills are double word aligned; but based on at least a pair of registers (i.e. multi-register or multiple word) simultaneously loaded as by Kahle's architecture (*lmw* – col. 4 ,lines 30-38) this *double-word* limitation would have been obvious in view of the rationale as set forth in claim 1.

As per claim 8, Kolson (in view of Kahle) discloses loading said paired register spills instructions from said memory stack back (by way of obviousness set forth in claim 6) into

matching register locations in each of said first register set and said second register set in parallel; but virtue of claim 2 and claim 1 (Note: loading of values into registers from claims 1-2, in view of claim 6 renders obvious the above limitation).

As per claim 9, Kolson (in view of Kahle) discloses a system for handling register spills in a CPU having parallel registers, comprising:

(a) a module for analyzing spill code generated by a register allocator to determine whether register spill instructions can be associated;

(b) a module for rewriting said register spill instructions as a parallel register spill instruction, if said register spill instructions can be associated;

(c) a module for configuring storage of associated register spills in memory in such a manner that said register spills can be loaded back into said registers in parallel based on said rewritten parallel register spill instruction;

the CPU having parallel registers, and the rewriting as one instruction for spill instruction loading into the parallel register as a parallel register instruction having been addressed as obvious in view of Kahle in the rationale as set forth in claim 1.

As per claims 10-15, refer to claims 2, 4-8 for corresponding rejection as set forth therein respectively.

As per claim 16, Kolson (in view of Kahle) discloses a system for handling register spills in a CPU having parallel registers, comprising:

(a) means for determining whether register spill instructions in spill code generated by a register allocator can be associated;

(b) means for determining if said register spill instructions can be associated, then rewriting said register spill instructions as a parallel register spill instruction;

(c) means for configuring, based on said rewritten parallel register spill instruction, storage of associated register spills in memory in such a manner that said register spills can be loaded back into said registers in parallel;

the CPU having parallel registers, and the rewriting as one instruction for spill instruction loading into the parallel register as a parallel register instruction having been addressed as obvious in view of Kahle in the rationale as set forth in claim 1.

As per claims 17-19, refer to claims 2, 7, 4 for corresponding rejection as set forth therein respectively.

As per claim 20, Kolson (in view of Kahle) a computer readable medium having computer readable program code embedded in the medium for handling register spills in a CPU having parallel registers, the computer readable program code including code for:

determining whether register spill instructions in spill code generated by a register allocator can be associated; for determining if said register spill instructions can be associated, then rewriting said register spill instructions as a parallel register spill instruction;

configuring, based on said rewritten parallel register spill instruction, storage of associated register spills in memory in such a manner that said register spills can be loaded back into said registers in parallel;

the CPU having parallel registers, and the rewriting as one instruction for spill instruction loading into the parallel register as a parallel register instruction having been addressed as obvious in view of Kahle in the rationale as set forth in claim 1.

As per claims 21-23, refer to claims 2, 7, 4 for corresponding rejection as set forth therein respectively.

Response to Arguments

5. Applicant's arguments filed 6/07/07 have been fully considered but they are not persuasive. Following are the Examiner's observation in regard thereto.

35 USC § 103(a) Rejection:

(A) Applicant has submitted that Kolson's loading of a variable into a register as proffered by the Office Action does not teach or suggest rewriting multiple spill instructions into a single parallel spill instruction (Appl. Rmrks pg. 17, bottom, pg. 18, top). It is noted that the Office Action clearly identifies which part of the claim Kolson does not teach, and therefrom provides the rationale using combined teachings to establish the obviousness of the limitation being identified. Apparently, the argument seems to ignore the grounds of the rejection as it has been laid out, and rather contends with expecting Kolson to **anticipate** the 'rewriting ... as a parallel register spill instruction' limitation. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See In re Keller, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); In re Merck & Co., 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

(B) Applicant has submitted that the Office Action contradicts in asserting a teaching of a *loading back* by Kolson then in later admitting that Kolson does not teach 'loading these into said parallel register in parallel'(Appl. Rmrks pg. 18, middle). As for the Office action, the following is an explanation of how the grounds of rejection have been set forth therein. The rejection discusses on Kolson's algorithm to extend the use of multiple registers, which entails a

configuration by which variables are loaded into registers, and because the 'loading back' has been not given merits beyond the loading connotation (see Claim Objection), the loading back limitation is considered anticipated in view of the above reconfiguring by Kolson's extended algorithm. The rejection identifies that Kolson does not explicit disclose (i) rewriting the reconfigured spill loading so that the rewritten instruction is a parallel loading instruction based on using a (ii) CPU having parallel register, a particular type of CPU that Kolson does not explicitly teach. The grounds of rejection has set forth identification of missing elements, and provided the suggestion of parallelism coming the multiple register exploiting endeavor by Kolson, which is taking advantage of concurrent variables loading while implementing the extended algorithm of Figure 4. In light of Kolson' s banks of registers and concurrent simultaneously loading as iteratively analyzed and configured in the above algorithm, the rejection has found Kahle with an alternate approach of providing floating point exploiting of registers and temporary storage to reconstruct a pipeline of similar register allocation to that of Kolson. Because Kahle suggests a complex architecture with layers of registers to reconstruct the register allocating scenario needed for a pipeline of concurrent processes performed within a architecture endowed with multiple processing units, the rejection has come up with the motivation to apply the concurrent loading by Kolson as suggested above, so that it borrows a complex CPU architecture by Kahle wherein floating point operations utilize a intermediate stage of reconfiguring pipeline-required data loading into successive register locations by means of first, a parallel register type of CPU and second, of rewriting (parts of a floating point allocation) into a intermediate register loading step. If the 103 rationale is detected as being improper (e.g. no motivation to combine, yielding inapposite effects), it is the responsibility of

the Applicant to convincingly point out where exactly in the rationale --to combine as set forth -- has failed to or cannot achieve acceptable result in order to fulfill the totality of 2 missing limitations being identified, interpreted and now paraphrased as follows: (a) the parallel register CPU, and (b) intermediate step of rewriting into a form of register reconfiguring whereby parallel register can be done via one spill instruction. The Office Action does not contradict with itself because it provides a clear approach by which the limitations as construed (via broad/reasonable interpretation based on a **objected to** claim language) have been identified as missing (from Kolson), then based on suggestion learned from analyzing the references, have then been rendered obvious. Notwithstanding that the cited portions are not for anticipating the missing elements as in a § 102 rejection, Applicant's argument is deemed insufficient to overcome the § 103 actual rejection; and further, Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims (i.e. objected to because of improper language) patentably distinguishes them from the references, and when dissecting any discrepancy against Kolson or Kahle individually, the Applicant cannot show non-obviousness by attacking references individually where the rejections are based on combinations of references. See In re Keller, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); In re Merck & Co., 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986).

(C) Applicant has submitted that the rest of the claims depend directly or indirectly from the above analysis against Kolson and Kahle, hence are allowable. This is to be referred back to section A and B above. In all, the claims stand rejected as set forth in the Office Action.

Conclusion

6. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (571) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai An can be reached on (571)272-3756.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence - please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Tuan A Vu
Patent Examiner,
Art Unit 2193
August 11, 2007